

FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFF	111111	111111	AAA	AAA
FFF	111111	111111	AAA	AAA
FFF	111111	111111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFFFFFFFFFFFFFF	111	111	AAA	AAA
FFFFFFFFFFFFFFF	111	111	AAA	AAA
FFFFFFFFFFFFFFF	111	111	AAA	AAA
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111111111	111111111	AAA	AAA
FFF	111111111	111111111	AAA	AAA
FFF	111111111	111111111	AAA	AAA

SC  
VO

.....

2

```

RRRRRRRR      WW      WW  VV      VV  BBBB88888
RRRRRRRR      WW      WW  VV      VV  BBBB88888
RR      RR      WW      WW  VV      VV  BB      BB
RR      RR      WW      WW  VV      VV  BB      BB
RR      RR      WW      WW  VV      VV  BB      BB
RR      RR      WW      WW  VV      VV  BB      BB
RRRRRRRR      WW      WW  VV      VV  BBBB88888
RRRRRRRR      WW      WW  VV      VV  BBBB88888
RR  RR      WW  WW  WW  VV      VV  BB      BB
RR  RR      WW  WW  WW  VV      VV  BB      BB
RR      RR      WWW  WWW  VV  VV  BB      BB
RR      RR      WWW  WWW  VV  VV  BB      BB
RR      RR      WW      WW      VV      BBBB88888
RR      RR      WW      WW      VV      BBBB88888

```

```

LL               IIIIII             SSSSSSSS
LL               IIIIII             SSSSSSSS
                II                  SS
LL              III                 SS
LL              III                 SS
LL              III                 SS
LL              III                 SS
LL              III                 SSSSSS
LL              III                 SSSSSS
LL              III                 SS
LL              III                 SS
LL              III                 SS
L L            III                 SSSSSS
LLLLLLLLLLLLLL IIIIIIII           SSSSSSSS
LLLLLLLLLLLLLL IIIIIIII           SSSSSSSS

```



```
1 0001 0 MODULE RWVB (
2 0002 0 LANGUAGE (BLISS32),
3 0003 0 IDENT = 'V04-000'
4 0004 0 ) =
5 0005 1 BEGIN
6 0006 1
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 * ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 * TRANSFERRED.
20 0020 1 *
21 0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 * CORPORATION.
24 0024 1 *
25 0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *
29 0029 1 *****
30 0030 1
31 0031 1 ++
32 0032 1
33 0033 1 FACILITY: F11ACP Structure Level 1
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1 This routine performs the window turn necessary to map a
38 0038 1 virtual I/O transfer which is not mapped by the current
39 0039 1 window. It also receives virtual I/O errors for bad block
40 0040 1 processing.
41 0041 1
42 0042 1 ENVIRONMENT:
43 0043 1
44 0044 1 STARLET operating system, including privileged system services
45 0045 1 and internal exec routines.
46 0046 1
47 0047 1 --
48 0048 1
49 0049 1
50 0050 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 7-Jan-1977 00:48
51 0051 1
52 0052 1 MODIFIED BY:
53 0053 1
54 0054 1 V03-001 ACG0320 Andrew C. Goldstein, 22-Mar-1983 12:27
55 0055 1 Change byte count handling to track IOPOST changes
56 0056 1
57 0057 1 A0101 ACG23542 Andrew C. Goldstein, 7-May-1979 13:36
```

RWVB  
V04-000

J 14  
16-Sep-1984 01:17:05  
14-Sep-1984 12:29:50

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11A.SRC]RWVB.B32;1

Page 2  
(1)

```
: 58      0058 1 | Check LBN of mapped VBN against volume size
: 59      0059 1 |
: 60      0060 1 | A0100 ACG00001 Andrew C. Goldstein, 10-Oct-1978 20:03
: 61      0061 1 | Previous revision history moved to F11A.REV
: 62      0062 1 | **
: 63      0063 1 |
: 64      0064 1 |
: 65      0065 1 | LIBRARY 'SYSS$LIBRARY:LIB.L32';
: 66      0066 1 | REQUIRE 'SRC$:FCPDEF.B32';
: 67      0381 1 |
: 68      0382 1 |
: 69      0383 1 | FORWARD ROUTINE
: 70      0384 1 | READ_WRITEVB, ! main read/write virtual handling
: 71      0385 1 | MARKBAD_FCB; ! mark bad block in FCB
```



```
73 0386 1 GLOBAL ROUTINE READ_WRITEVB =
74 0387 1
75 0388 1 ++
76 0389 1
77 0390 1 FUNCTIONAL DESCRIPTION:
78 0391 1
79 0392 1 This routine performs the window turn necessary to map a
80 0393 1 virtual I/O transfer which is not mapped by the current
81 0394 1 window. It also receives virtual I/O errors for bad block
82 0395 1 processing. These are presently simply returned to the user.
83 C-96 1
84 0397 1 CALLING SEQUENCE:
85 0398 1 READ_WRITEVB ( )
86 0399 1
87 0400 1 INPUT PARAMETERS:
88 0401 1 NONE
89 0402 1
90 0403 1 IMPLICIT INPUTS:
91 0404 1 IO_PACKET: I/O packet of request
92 0405 1
93 0406 1 OUTPUT PARAMETERS:
94 0407 1 NONE
95 0408 1
96 0409 1 IMPLICIT OUTPUTS:
97 0410 1 NONE
98 0411 1
99 0412 1 ROUTINE VALUE:
100 0413 1 1 if request queued to driver
101 0414 1 0 if error
102 0415 1
103 0416 1 SIDE EFFECTS:
104 0417 1 window turned
105 0418 1 request queued to driver if mapped
106 0419 1
107 0420 1 --
108 0421 1
109 0422 2 BEGIN
110 0423 2
111 0424 2 LOCAL
112 0425 2 PACKET : REF BBLOCK, ! pointer to I/O packet
113 0426 2 WINDOW : REF BBLOCK, ! file window
114 0427 2 FCB : REF BBLOCK, ! file FCB
115 0428 2 BLOCK_COUNT, ! number of blocks in transfer
116 0429 2 UNMAPPED, ! number of blocks not mapped
117 0430 2 MODE, ! mode (read/write) of transfer
118 0431 2 VBN, ! starting VBN of transfer
119 0432 2 LBN, ! translated LBN
120 0433 2 LAST_LBN; ! highest LBN touched by operation
121 0434 2
122 0435 2 EXTERNAL
123 0436 2 USER_STATUS : VECTOR, ! user I/O status block
124 0437 2 IO_PACKET : REF BBLOCK, ! I/O request packet
125 0438 2 CURRENT_VCB : REF BBLOCK, ! VCB of volume in use
126 0439 2 CURRENT_UCB : REF BBLOCK; ! UCB of volume in use
127 0440 2
128 0441 2 EXTERNAL ROUTINE
129 0442 2 MAP_VBN, ! map and turn window
```

```
130 0443 2      REQUEUE REQ,
131 0444 2      SCAN_BADLOG;
132 0445 2
133 0446 2
134 0447 2      ! Extract the request parameters from the I/O packet. Compute VBN and LBN
135 0448 2      ! of the next block to be transferred.
136 0449 2
137 0450 2
138 0451 2      PACKET = .IO_PACKET;
139 0452 2      WINDOW = .PACKET[IRPSL_WIND];
140 0453 2      BLOCK_COUNT = (.PACKET[IRPSW_BCNT]+511) / 512;
141 0454 2      VBN = .PACKET[IRPSL_SEGVBN];
142 0455 2
143 0456 2      IF .VBN EQL 0 THEN ERR_EXIT (SS$_BADPARAM);
144 0457 2
145 0458 2
146 0459 2      ! Attempt to map the request. If the map fails, report
147 0460 2      ! failure. Else requeue the request to the driver.
148 0461 2
149 0462 2
150 0463 2      LBN = MAP_VBN (.VBN, .WINDOW, .BLOCK_COUNT, UNMAPPED);
151 0464 2      IF .LBN EQL -1 THEN ERR_EXIT (SS$_ENDOFFILE);
152 0465 2
153 0466 2      IF .PACKET[IRPSV_VIRTUAL]
154 0467 2      THEN
155 0468 2          BEGIN
156 0469 3          LAST_LBN = .LBN + (.BLOCK_COUNT - .UNMAPPED - 1);
157 0470 3          IF .LBN GEQU .CURRENT_UCB[UCBSL_MAXBLOCK]
158 0471 3          OR .LAST_LBN GEQU .CURRENT_UCB[UCBSL_MAXBLOCK]
159 0472 3          THEN ERR_EXIT (SS$_ILLBLKNOM);
160 0473 3          KERNEL_CALL (REQUEUE_REQ, .PACKET, .LBN, .UNMAPPED);
161 0474 3          RETURN 1;
162 0475 3          END
163 0476 3
164 0477 3      ! If the virtual bit is not set, this is an I/O error on a file sent here
165 0478 3      ! for bad block processing. If the error is a parity, format, or datacheck
166 0479 3      ! error, we set the bad block bit in the FCB of the file and enter the
167 0480 3      ! block in question into the volume's bad block log. Note that we do not
168 0481 3      ! do this on errors on the volume's reserved files, which are not subject
169 0482 3      ! to dynamic bad block processing.
170 0483 3
171 0484 3
172 0485 2      ELSE
173 0486 2          BEGIN
174 0487 3          FCB = .WINDOW[WCBSL_FCB];
175 0488 3
176 0489 4          IF (
177 0490 4              .(PACKET[IRPSL_IOST1])<0,16> EQL SS$_PARITY
178 0491 4              OR .(PACKET[IRPSL_IOST1])<0,16> EQL SS$_DATACHECK
179 0492 4              OR .(PACKET[IRPSL_IOST1])<0,16> EQL SS$_FORMAT
180 0493 4              )
181 0494 4          AND (
182 0495 4              .FCB[FCBSW_FID_NUM] GTRU .CURRENT_VCB[VCBSB_RESFILES]
183 0496 5              OR (.CURRENT_VCB[VCBSV_EXTFID]
184 0497 5                  AND .FCB[FCBSB_FID_NMX] NEQ 0)
185 0498 4              )
186 0499 3          THEN
```



```
! end of routine READ_WRITEVB
```

```
CHMU      #220                                : 0472
RET                                              :
PUSHL     UNMAPPED                            : 0473
```

					00FC	00000	
		57	000000000G	9F	9E	00002	
		5E		04	C2	00009	
		53	0000G	CF	D0	0000C	
		55	18	A3	D0	00011	
		50	32	A3	3C	00015	
		50	01FF	C0	9E	00019	
52		50	00000200	8F	C7	0001E	
		56	48	A3	D0	00026	
				03	12	0002A	
				14	BF	0002C	
					04	0002E	
			4004	8F	BB	0002F	1\$:
				55	DD	00033	
				56	DD	00035	
	0000G	CF		04	FB	00037	
		54		50	D0	0003C	
	FFFFFFFF	8F		54	D1	0003F	
				05	12	00046	
			0870	8F	BF	00048	
					04	0004C	
33	2A	A3		04	E1	0004D	2\$:
		52		6E	C2	00052	
		51	FF A244	9E	00055		
		50	0000G	CF	D0	0005A	
	00B0	C0		54	D1	0005F	
				07	1E	00064	
	00B0	C0		51	D1	00066	
				05	1F	0006B	
			00DC	8F	BF	0006D	3\$:
					04	00071	
				6E	DD	00072	4\$:

			18	BB	00074	PUSHR	#^M<R3,R4>	:	
			03	DD	00076	PUSHL	#3	:	
			5E	DD	00078	PUSHL	SP	:	
		0000G	CF	9F	0007A	PUSHAB	REQUEUE REQ	:	
	67		06	FB	0007E	CALLS	#6, SYS\$CMKRNL	:	
	50		01	D0	00081	MOVL	#1, R0	:	0486
				04	00084	RET		:	
	52	18	A5	D0	00085	5\$:	MOVL 24(WINDOW), FCB	:	0487
01F4	8F	38	A3	B1	00089		CMPW 56(PACKET), #500	:	0490
			10	13	0008F		BEQL 6\$	:	
005C	8F	38	A3	B1	00091		CMPW 56(PACKET), #92	:	0491
			08	13	00097		BEQL 6\$	:	
00BC	8F	38	A3	B1	00099		CMPW 56(PACKET), #188	:	0492
			44	12	0009F		BNEQ 9\$	:	
	50	0000G	CF	D0	000A1	6\$:	MOVL CURRENT_VCB, R0	:	0495
	51	4F	A0	9A	000A6		MOVZBL 79(R0), R1	:	
24	A2		51	B1	000AA		CMPW R1, 36(FCB)	:	
			0A	1F	000AE		BLSSU 7\$	:	
30	0B	A0	05	E1	000B0		BBC #5, 11(R0), 9\$	:	0496
			29	A2	95	000B5	TSTB 41(FCB)	:	0497
			2B	13	000B8		BEQL 9\$	:	
			52	DD	000BA	7\$:	PUSHL FCB	:	0501
			01	DD	000BC		PUSHL #1	:	
			5E	DD	000BE		PUSHL SP	:	
		0000V	CF	9F	000C0		PUSHAB MARKBAD_FCB	:	
	67		04	FB	000C4		CALLS #4, SYS\$CMKRNL	:	
	50		01	D0	000C7		MOVL #1, MODE	:	0502
0B	20	A3	00	ED	000CA		CMPZV #0, #6, 32(PACKET), #11	:	0503
	06		03	12	000D0		BNEQ 8\$	:	
	50		02	D0	000D2		MOVL #2, MODE	:	0504
			7E	D4	000D5	8\$:	CLRL -(SP)	:	0505
			50	DD	000D7		PUSHL MODE	:	
			54	DD	000D9		PUSHL LBN	:	
			56	DD	000DB		PUSHL VBN	:	
		24	A2	9F	000DD		PUSHAB 36(FCB)	:	
0000G	CF		05	FB	000E0		CALLS #5, SCAN_BADLOG	:	
0000G	CF	38	A3	7D	000E5	9\$:	MOVQ 56(PACKET), USER_STATUS	:	0507
			50	D4	000EB		CLRL R0	:	0509
			04	000ED			RET	:	0512

; Routine Size: 238 bytes, Routine Base: \$CODE\$ + 0000



```

: 201 0513 1 GLOBAL ROUTINE MARKBAD_FCB (FCB) =
: 202 0514 1
: 203 0515 1 ++
: 204 0516 1
: 205 0517 1 FUNCTIONAL DESCRIPTION:
: 206 0518 1
: 207 0519 1 This routine set the bad block bit in the indicated FCB.
: 208 0520 1
: 209 0521 1
: 210 0522 1 CALLING SEQUENCE:
: 211 0523 1 MARKBAD_FCB (ARG1)
: 212 0524 1
: 213 0525 1 INPUT PARAMETERS:
: 214 0526 1 ARG1: address of FCB
: 215 0527 1
: 216 0528 1 IMPLICIT INPUTS:
: 217 0529 1 NONE
: 218 0530 1
: 219 0531 1 OUTPUT PARAMETERS:
: 220 0532 1 NONE
: 221 0533 1
: 222 0534 1 IMPLICIT OUTPUTS:
: 223 0535 1 NONE
: 224 0536 1
: 225 0537 1 ROUTINE VALUE:
: 226 0538 1 1
: 227 0539 1
: 228 0540 1 SIDE EFFECTS:
: 229 0541 1 bad bit set in FCB
: 230 0542 1
: 231 0543 1 --
: 232 0544 1
: 233 0545 2 BEGIN
: 234 0546 2
: 235 0547 2 MAP
: 236 0548 2 FCB : REF BBLOCK; ! FCB argument
: 237 0549 2
: 238 0550 2
: 239 0551 2 FCB[FCB$V_BADBLK] = 1;
: 240 0552 2
: 241 0553 2 RETURN 1;
: 242 0554 2
: 243 0555 1 END; ! end of routine MARKBAD_FCB
```

```

          0000 00000
          22 50 04 AC D0 00002
          AO 04 88 00006
          50 01 D0 0000A
          04 0000D
```

```

.ENTRY MARKBAD_FCB, Save nothing
MOVL FCB, R0
BISB2 #4, 34(R0)
MOVL #1, R0
RET
```

```

: 0513
: 0551
: 0553
: 0555
```

; Routine Size: 14 bytes, Routine Base: \$CODE\$ + 00EE



RWVB  
V04-000

C 15  
16-Sep-1984 01:17:05  
14-Sep-1984 12:29:50

VAX-11 BLISS-32 V4.0-742  
DISK\$VMSMASTER:[F11A.SRC]RWVB.B32;1 Page 8 (3)

: 244 0556 1  
: 245 0557 1 END  
: 246 0558 0 ELUDOM

PSECT SUMMARY

: Name Bytes Attributes  
: \$CODE\$ 252 NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

: File Total Symbols Loaded Percent Pages Mapped Processing Time  
: \_\$255\$DUA28:[SYSLIB]LIB.L32;1 18619 23 0 1000 00:01.9

COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:RWVB/OBJ=OBJ\$:RWVB MSRC\$:RWVB/UPDATE=(ENH\$:RWVB)

: Size: 252 code + 0 data bytes  
: Run Time: 00:09.2  
: Elapsed Time: 00:24.8  
: Lines/CPU Min: 3643  
: Lexemes/CPU-Min: 14128  
: Memory Used: 118 pages  
: Compilation Complete



0166 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

